# SQL and PySpark

## Select Columns

### SQL

```
SELECT column1, column2 FROM table;
```

### PySpark

```
df.select("column1", "column2")
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Filter Rows

| SQL | PySpark |
|---|---|
| *SELECT * FROM table WHERE condition;* | *df.filter("condition")* |

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Aggregate Functions

### SQL

```
SELECT AVG(column) FROM table;
```

### PySpark

```
df.select(F.avg("column"))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Group By

### SQL

```
SELECT column, COUNT(*) FROM table
GROUP BY column;
```

### PySpark

```
df.groupBy("column").count()
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Order By

### SQL

*SELECT * FROM table ORDER BY column ASC;*

### PySpark

*df.orderBy("column", ascending=True)*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Join

### SQL

```
SELECT * FROM table1 JOIN table2 ON
table1.id = table2.id;
```

### PySpark

```
df1.join(df2, df1.id == df2.id)
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Union

### SQL

```
SELECT * FROM table1 UNION SELECT *
FROM table2;
```

### PySpark

```
df1.union(df2)
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Limit

### SQL

```
SELECT * FROM table LIMIT 100;
```

### PySpark

```
df.limit(100)
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Distinct Values

### SQL

```
SELECT DISTINCT column FROM table;
```

### PySpark

```
df.select("column").distinct()
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Adding a New Column

### SQL

*SELECT *, (column1 + column2) AS new_column FROM table;*

### PySpark

*df.withColumn("new_column", F.col("column1") + F.col("column2"))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Column Alias

### SQL

```
SELECT column AS alias_name FROM table;
```

### PySpark

```
df.select(F.col("column").alias("alias_name"))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Filtering on Multiple Conditions

### SQL

*SELECT * FROM table WHERE condition1 AND condition2;*

### PySpark

*df.filter((F.col("condition1")) & (F.col("condition2")))*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Subquery

### SQL

```
SELECT * FROM (SELECT * FROM table
WHERE condition) AS subquery;
```

### PySpark

```
df.filter("condition").alias("subquery")
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Between

### SQL

```
SELECT * FROM table WHERE column BETWEEN val1 AND val2;
```

### PySpark

```
df.filter(F.col("column").between("val1", "val2"))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Like

### SQL

*SELECT * FROM table WHERE column LIKE pattern;*

### PySpark

*df.filter(F.col("column").like("pattern"))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Case When

### SQL

```
SELECT CASE WHEN condition THEN result1
ELSE result2 END FROM table;
```

### PySpark

```
df.select(F.when(F.col("condition"),
"result1").otherwise("result2"))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Cast Data Type

### SQL

*SELECT CAST(column AS datatype) FROM table;*

### PySpark

*df.select(F.col("column").cast("datatype"))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Count Distinct

### SQL

*SELECT COUNT(DISTINCT column) FROM table;*

### PySpark

*df.select(F.countDistinct("column"))*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Substring

### SQL

```
SELECT SUBSTRING(column, start, length)
FROM table;
```

### PySpark

```
df.select(F.substring("column", start, length))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Concatenate Columns

### SQL

*SELECT CONCAT(column1, column2) AS new_column FROM table;*

### PySpark

*df.withColumn("new_column", F.concat(F.col("column1"), F.col("column2")))*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Average Over Partition

### SQL

```
SELECT AVG(column) OVER (PARTITION BY column2) FROM table;
```

### PySpark

```
df.withColumn("avg", F.avg("column").over(Window.partitionBy("column2")))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Sum Over Partition

### SQL

```
SELECT SUM(column) OVER (PARTITION BY column2) FROM table;
```

### PySpark

```
df.withColumn("sum", F.sum("column").over(Window.partitionBy("column2")))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Lead Function

### SQL

```
SELECT LEAD(column, 1) OVER (ORDER BY column2) FROM table;
```

### PySpark

```
df.withColumn("lead", F.lead("column", 1).over(Window.orderBy("column2")))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Lag Function

### SQL

```
SELECT LAG(column, 1) OVER (ORDER BY
column2) FROM table;
```

### PySpark

```
df.withColumn("lag", F.lag("column",
1).over(Window.orderBy("column2")))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Row Count

### SQL

*SELECT COUNT(*) FROM table;*

### PySpark

*df.count()*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Drop Column

### SQL

*ALTER TABLE table DROP COLUMN column;*
*(Not directly in SELECT)*

### PySpark

*df.drop("column")*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Rename Column

### SQL

*ALTER TABLE table RENAME COLUMN column1 TO column2; (Not directly in SELECT)*

### PySpark

*df.withColumnRenamed("column1", "column2")*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Change Column Type

### SQL

*ALTER TABLE table ALTER COLUMN column TYPE new_type; (Not directly in SELECT)*

### PySpark

*df.withColumn("column", df["column"].cast("new_type"))*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Creating a Table from Select

### SQL

*CREATE TABLE new_table AS SELECT * FROM table;*

### PySpark

*(df.write.format("parquet").saveAsTable("new_table"))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Inserting Selected Data into Table

### SQL

*INSERT INTO table2 SELECT * FROM table1;*

### PySpark

*(df1.write.insertInto("table2"))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Creating a Table with Specific Columns

### SQL

```
CREATE TABLE new_table AS SELECT
column1, column2 FROM table;
```

### PySpark

```
(df.select("column1",
"column2").write.format("parquet").saveAsTable(
"new_table"))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Aggregate with Alias

### SQL

*SELECT column, COUNT(*) AS count FROM table GROUP BY column;*

### PySpark

*df.groupBy("column").agg(F.count("*").alias("count"))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Nested Subquery

### SQL

*SELECT * FROM (SELECT * FROM table WHERE condition) sub WHERE sub.condition2;*

### PySpark

*df.filter("condition").alias("sub").filter("sub.condition2")*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Multiple Joins

### SQL

```
SELECT * FROM table1 JOIN table2 ON
table1.id = table2.id JOIN table3 ON
table1.id = table3.id;
```

### PySpark

```
df1.join(df2, "id").join(df3, "id")
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Cross Join

### SQL

```
SELECT * FROM table1 CROSS JOIN table2;
```

### PySpark

```
df1.crossJoin(df2)
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Group By Having Count Greater Than

### SQL

*SELECT column, COUNT(*) FROM table GROUP BY column HAVING COUNT(*) > 1;*

### PySpark

*df.groupBy("column").count().filter(F.col("count") > 1)*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Alias for Table in Join

### SQL

```
SELECT t1.* FROM table1 t1 JOIN table2 t2
ON t1.id = t2.id;
```

### PySpark

```
df1.alias("t1").join(df2.alias("t2"), F.col("t1.id") ==
F.col("t2.id"))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Selecting from Multiple Tables

### SQL

*SELECT t1.column, t2.column FROM table1 t1, table2 t2 WHERE t1.id = t2.id;*

### PySpark

*df1.join(df2, df1.id == df2.id).select(df1.column, df2.column)*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Case When with Multiple Conditions

### SQL

*SELECT CASE WHEN condition THEN 'value1' WHEN condition2 THEN 'value2' ELSE 'value3' END FROM table;*

### PySpark

*df.select(F.when(F.col("condition"), "value1").when(F.col("condition2"), "value2").otherwise("value3"))*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Extracting Date Parts

### SQL

```
SELECT EXTRACT(YEAR FROM date_column)
FROM table;
```

### PySpark

```
df.select(F.year(F.col("date_column")))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Inequality Filtering

### SQL

```
SELECT * FROM table WHERE column != 'value';
```

### PySpark

```
df.filter(df.column != 'value')
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## In List

### SQL

*SELECT * FROM table WHERE column IN ('value1', 'value2');*

### PySpark

*df.filter(df.column.isin('value1', 'value2'))*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Not In List

### SQL

```
SELECT * FROM table WHERE column NOT IN ('value1', 'value2');
```

### PySpark

```
df.filter(~df.column.isin('value1', 'value2'))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Null Values

### SQL

*SELECT * FROM table WHERE column IS NULL;*

### PySpark

*df.filter(df.column.isNull())*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Not Null Values

### SQL

*SELECT * FROM table WHERE column IS NOT NULL;*

### PySpark

*df.filter(df.column.isNotNull())*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## String Upper Case

### SQL

```
SELECT UPPER(column) FROM table;
```

### PySpark

```
df.select(F.upper(df.column))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## String Lower Case

### SQL

```
SELECT LOWER(column) FROM table;
```

### PySpark

```
df.select(F.lower(df.column))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## String Length

### SQL

```
SELECT LENGTH(column) FROM table;
```

### PySpark

```
df.select(F.length(df.column))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Trim String

### SQL

```
SELECT TRIM(column) FROM table;
```

### PySpark

```
df.select(F.trim(df.column))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Left Trim String

### SQL

*SELECT LTRIM(column) FROM table;*

### PySpark

*df.select(F.ltrim(df.column))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Right Trim String

### SQL

```sql
SELECT RTRIM(column) FROM table;
```

### PySpark

```python
df.select(F.rtrim(df.column))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## String Replace

### SQL

*SELECT REPLACE(column, 'find', 'replace')
FROM table;*

### PySpark

*df.select(F.regexp_replace(df.column, 'find', 'replace'))*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Substring Index

### SQL

*SELECT SUBSTRING_INDEX(column, 'delim', count) FROM table;*

### PySpark

*df.select(F.expr("split(column, 'delim')[count-1]"))*
*(Assuming 1-based index)*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Date Difference

### SQL

```
SELECT DATEDIFF('date1', 'date2') FROM table;
```

### PySpark

```
df.select(F.datediff(F.col('date1'), F.col('date2')))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Add Months to Date

### SQL

```
SELECT ADD_MONTHS(date_column,
num_months) FROM table;
```

### PySpark

```
df.select(F.add_months(df.date_column,
num_months))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## First Value in Group

### SQL

```
SELECT FIRST_VALUE(column) OVER
(PARTITION BY column2) FROM table;
```

### PySpark

```
df.withColumn("first_val",
F.first("column").over(Window.partitionBy("colum
n2")))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Last Value in Group

### SQL

*SELECT LAST_VALUE(column) OVER (PARTITION BY column2) FROM table;*

### PySpark

*df.withColumn("last_val", F.last("column").over(Window.partitionBy("column2")))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Row Number Over Partition

### SQL

*SELECT ROW_NUMBER() OVER (PARTITION BY column ORDER BY column) FROM table;*

### PySpark

*df.withColumn("row_num", F.row_number().over(Window.partitionBy("column").orderBy("column")))*

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Rank Over Partition

### SQL

```
SELECT RANK() OVER (PARTITION BY
column ORDER BY column) FROM table;
```

### PySpark

```
df.withColumn("rank",
F.rank().over(Window.partitionBy("column").order
By("column")))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Dense Rank Over Partition

### SQL

*SELECT DENSE_RANK() OVER (PARTITION BY column ORDER BY column) FROM table;*

### PySpark

*df.withColumn("dense_rank", F.dense_rank().over(Window.partitionBy("column").orderBy("column")))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Count Rows

### SQL

*SELECT COUNT(*) FROM table;*

### PySpark

*df.count()*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Mathematical Operations

### SQL

```
SELECT column1 + column2 FROM table;
```

### PySpark

```
df.select(F.col("column1") + F.col("column2"))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## String Concatenation

### SQL

```
SELECT column1 | column2 AS new_column
FROM table;
```

### PySpark

```
df.withColumn("new_column", F.concat_ws("|",
F.col("column1"), F.col("column2")))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Find Minimum Value

### SQL

```
SELECT MIN(column) FROM table;
```

### PySpark

```
df.select(F.min("column"))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Find Maximum Value

### SQL

*SELECT MAX(column) FROM table;*

### PySpark

*df.select(F.max("column"))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Removing Duplicates

### SQL

*SELECT DISTINCT * FROM table;*

### PySpark

*df.distinct()*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Left Join

### SQL

```
SELECT * FROM table1 LEFT JOIN table2 ON table1.id = table2.id;
```

### PySpark

```
df1.join(df2, df1.id == df2.id, "left")
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Right Join

### SQL

```
SELECT * FROM table1 RIGHT JOIN table2
ON table1.id = table2.id;
```

### PySpark

```
df1.join(df2, df1.id == df2.id, "right")
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Full Outer Join

### SQL

*SELECT * FROM table1 FULL OUTER JOIN table2 ON table1.id = table2.id;*

### PySpark

*df1.join(df2, df1.id == df2.id, "outer")*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Group By with Having

### SQL

```
SELECT column, COUNT(*) FROM table
GROUP BY column HAVING COUNT(*) > 10;
```

### PySpark

```
df.groupBy("column").count().filter(F.col("count")
> 10)
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Round Decimal Values

### SQL

```
SELECT ROUND(column, 2) FROM table;
```

### PySpark

```
df.select(F.round("column", 2))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Get Current Date

### SQL

```
SELECT CURRENT_DATE();
```

### PySpark

```
df.select(F.current_date())
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Date Addition

### SQL

```
SELECT DATE_ADD(date_column, 10) FROM table;
```

### PySpark

```
df.select(F.date_add(F.col("date_column"), 10))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Date Subtraction

### SQL

*SELECT DATE_SUB(date_column, 10) FROM table;*

### PySpark

*df.select(F.date_sub(F.col("date_column"), 10))*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Extract Year from Date

### SQL

```
SELECT YEAR(date_column) FROM table;
```

### PySpark

```
df.select(F.year(F.col("date_column")))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Extract Month from Date

### SQL

```
SELECT MONTH(date_column) FROM table;
```

### PySpark

```
df.select(F.month(F.col("date_column")))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Extract Day from Date

### SQL

```
SELECT DAY(date_column) FROM table;
```

### PySpark

```
df.select(F.dayofmonth(F.col("date_column")))
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Sorting Descending

### SQL

*SELECT * FROM table ORDER BY column DESC;*

### PySpark

*df.orderBy(F.col("column").desc())*

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Group By Multiple Columns

### SQL

```
SELECT col1, col2, COUNT(*) FROM table
GROUP BY col1, col2;
```

### PySpark

```
df.groupBy("col1", "col2").count()
```

Shwetank Singh

**GritSetGrow - GSGLearn.com**

# SQL and PySpark

## Conditional Column Update

### SQL

```
UPDATE table SET column1 = CASE WHEN
condition THEN 'value1' ELSE 'value2' END;
```

### PySpark

```
df.withColumn("column1",
F.when(F.col("condition"),
"value1").otherwise("value2"))
```

Shwetank Singh
**GritSetGrow - GSGLearn.com**